

## CLAIMS

- [c1] 1. A method in a computer system for implementing a circular buffer, comprising:  
storing in forwarding words, located past an end of the buffer, pointers to locations at the  
other end of the buffer;  
enabling forwarding in the pointers; and  
when a forwarding word is accessed, directing the access to the pointed to location at the  
other end of the buffer.
- [c2] 2. The method of claim 1 wherein the buffer is pointed to by a write pointer whose  
value modulo a size of the buffer indicates the starting position for storing data in the buffer.
- [c3] 3. The method of claim 1 wherein the buffer is pointed to by a read pointer whose  
value modulo a size of the buffer indicates the starting position for reading data from the buffer.
- [c4] 4. The method of claim 1 wherein the access is a read.
- [c5] 5. The method of claim 1 wherein the access is a write.
- [c6] 6. The method of claim 1 wherein the access is using a pointer.
- [c7] 7. The method of claim 6 wherein the pointer is a write pointer.
- [c8] 8. The method of claim 6 wherein the pointer is a read pointer.
- [c9] 9. The method of claim 6 wherein the pointer has a synchronization access mode.
- [c10] 10. The method of claim 9 wherein the synchronization mode is sync.
- [c11] 11. The method of claim 9 wherein the synchronization mode is normal.

- [c12]            12.     The method of claim 9 wherein the synchronization mode can be set.
- [c13]            13.     The method of claim 1 wherein the access does not include code for detecting the end of the buffer.
- [c14]            14.     The method of claim 1 further comprising:  
when adding data to the buffer,  
                 receiving an indication of data to be written, the data having a size;  
                 fetching a write pointer;  
                 adding an indication of the size of the data to the write pointer; and  
                 copying the data into the buffer starting at a location indicated by the fetched  
                 write pointer.
- [c15]            15.     The method of claim 14 wherein the fetching and adding includes executing a  
fetch and add operation.
- [c16]            16.     The method of claim 14 wherein when the copying would occur in a word located  
past an end of the buffer, the copying automatically circles to the other end of the buffer.
- [c17]            17.     The method of claim 14 wherein the adding includes calculating a modulo of a  
sum of the addition and a size of the buffer.
- [c18]            18.     The method of claim 1 further comprising:  
when reading data from the buffer,  
                 receiving an indication of a location where read data is to be stored;  
                 fetching a read pointer;  
                 reading a size of the data to be read from the buffer; and  
                 copying data from the buffer to the indicated location.

- [c19]            19.     The method of claim 18 further comprising setting the read pointer to a sum of the read pointer and the size of the data modulo a size of the buffer.
- [c20]            20.     The method of claim 18 wherein the read pointer is accessed with a synchronization access mode of sync.
- [c21]            21.     The method of claim 18 wherein the data is read from the buffer using an access control mode of the read pointer.
- [c22]            22.     The method of claim 1 wherein when the access has a synchronization access mode of sync, read access to a location in the buffer is permitted only when the location is full.
- [c23]            23.     The method of claim 22 wherein after the read access, the location is set to empty.
- [c24]            24.     The method of claim 1 wherein when the access has a synchronization access mode of sync, write access to a location in the buffer is permitted only when the location is empty.
- [c25]            25.     The method of claim 24 wherein after the write access, the location is set to full.
- [c26]            26.     The method of claim 1 including storing a pointer to an invalid location in a location adjacent to the forwarding words with forwarding of that location enabled so that when the location adjacent to the forwarding words is accessed, an exception is raised.
- [c27]            27.     The method of claim 1 wherein the buffer is accessed by multiple readers and writers.
- [c28]            28.     The method of claim 1 wherein the buffer is accessed by multiple producers.
- [c29]            29.     The method of claim 1 wherein the buffer is accessed by multiple consumers.

- [c30] 30. A method in a computer system for detecting access to a memory location adjacent to a data structure, the method comprising:  
storing a pointer to an invalid memory location in the memory location;  
enabling forwarding for the memory location; and  
when access to the invalid memory location through the memory location raises an exception, indicating that the memory location adjacent to the data structure has been accessed.
- [c31] 31. The method of claim 30 wherein when speculative loads are enabled, the indicating includes setting a poison bit in a destination register when the access is a load from the memory location.
- [c32] 32. The method of claim 31 wherein the exception is raised when a value in the invalid memory location is used.
- [c33] 33. The method of claim 30 wherein the access does not disable the forwarding.
- [c34] 34. The method of claim 30 wherein the exception is a data protection exception.
- [c35] 35. The method of claim 30 wherein the exception causes a trap to occur.
- [c36] 36. A system for implementing a circular buffer, comprising:  
a component that stores in forwarding words located past an end of a buffer pointers to locations at the other end of the buffer and enables forwarding in the pointers;  
a component that accesses the buffer; and  
a component that, when a forwarding word is accessed, directs the access to the pointed to location at the other end of the buffer.
- [c37] 37. The system of claim 36 wherein the buffer is accessed by multiple readers and writers.

- [c38]            38.    The system of claim 36 wherein the buffer is pointed to by a read pointer whose value modulo a size of the buffer indicates the starting position for reading data from the buffer.
- [c39]            39.    The system of claim 36 wherein the access is a read.
- [c40]            40.    The system of claim 36 wherein the access is a write.
- [c41]            41.    The system of claim 36 wherein the access is using a pointer.
- [c42]            42.    The system of claim 41 wherein the pointer is a write pointer.
- [c43]            43.    The system of claim 41 wherein the pointer is a read pointer.
- [c44]            44.    The system of claim 41 wherein the pointer has a synchronization access mode.
- [c45]            45.    The system of claim 44 wherein the synchronization mode is sync.
- [c46]            46.    The system of claim 44 wherein the synchronization mode is normal.
- [c47]            47.    The system of claim 44 wherein the synchronization mode can be set.
- [c48]            48.    The system of claim 36 wherein the access does not include code for detecting the end of the buffer.
- [c49]            49.    A computer-readable medium for implementing a circular buffer, comprising:  
a buffer with storage locations, the buffer having a beginning and an end;  
a forwarding word adjacent to the end of the buffer; and  
a pointer in the forwarding word pointing to the beginning of the buffer.
- [c50]            50.    The computer-readable medium of claim 49 further comprising multiple forwarding words wherein each forwarding word has a pointer to a storage location.

- [c51]            51.    The computer-readable medium of claim 50 wherein each forwarding word has forwarding enabled.
- [c52]            52.    The computer-readable medium of claim 49 wherein each forwarding word has forwarding enabled.
- [c53]            53.    The computer-readable medium of claim 49 further comprising a read pointer.
- [c54]            54.    The computer-readable medium of claim 53 wherein the value of the read pointer modulo a size of the buffer indicates a starting position for reading data from the buffer.
- [c55]            55.    The computer-readable medium of claim 49 further comprising a write pointer.
- [c56]            56.    The computer-readable medium of claim 55 wherein the value of the write pointer modulo a size of the buffer indicates a starting position for storing data in the buffer.
- [c57]            57.    A system for detecting access to a memory location adjacent to a data structure, the system comprising:  
                  a component that stores a pointer to an invalid memory location in the memory location;  
                  a component that enables forwarding for the memory location; and  
                  a component that, when access to the invalid memory location through the memory location raises an exception, indicates that the memory location adjacent to the data structure has been accessed.
- [c58]            58.    The system of claim 57 wherein when speculative loads are enabled, the indicating includes setting a poison bit in a destination register when the access is a load from that memory location.
- [c59]            59.    The system of claim 58 wherein the exception is raised when a value in the invalid memory location is used.

- [c60]            60.    The system of claim 57 wherein the access does not disable the forwarding.
- [c61]            61.    The system of claim 57 wherein the exception is a data protection exception.
- [c62]            62.    The system of claim 57 wherein the exception causes a trap to occur.